FOR COMMODORE 64

FONT DESIGN KIT WITH COLOR CONTROL

FONTROL

Send Fonts to:
EPSON FX-80, C-ITOH8510 (Prowriter)

FROM

LINIUM TECHNOLOGY, INC.

LINIUM TECHNOLOGY, INC. IS A WHOLLY OWNED SUBSIDIARY OF AMERICAN SOFTWARE TECHNOLOGY, INC. NASDQ SYMBOL: ASTK

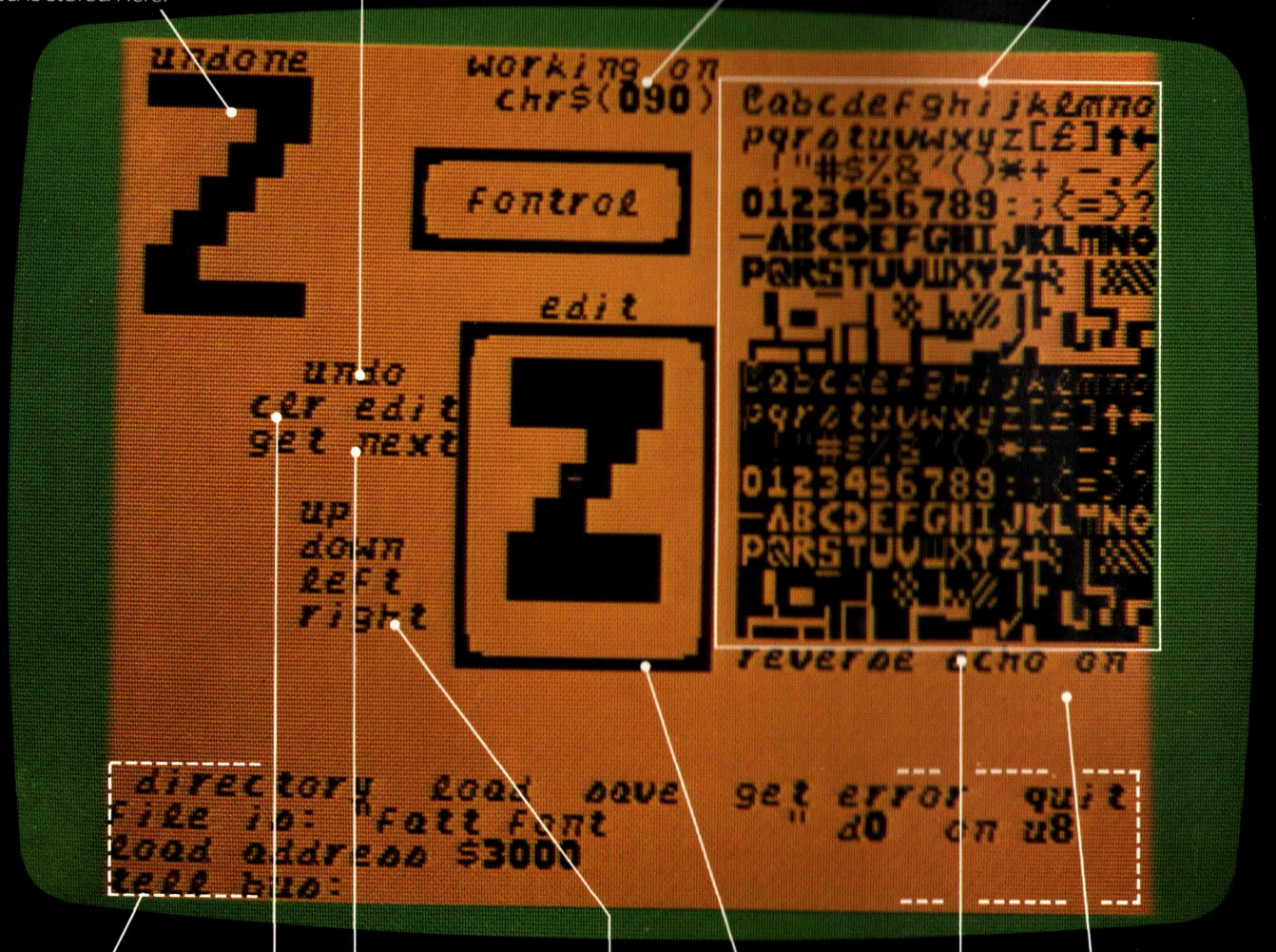
The editing display: New fonts are built here, old fonts retreaded. Everything you need is here, point at it to make it happen.[1]

Undone Window
So you'll know what undo is gonna do if done (what fun). The previous version of the character being edited is stored here.

Undo
Restores a character to its pre-edit version (Oops command).

Working On
This number is the ascii or screen poke value. Set numbers are: 1:Upper/Graphics, 2:Upper/Lower

Character Selection
The used font lot, pick them up here.[2]



File and Device Control Window
No Computer is an Island? Send a message in an IEEE bottle.[4]

Get Next Character
If you'd rather do it in order.

Edit
It all started here, this is your magnifying glass.

Descending Characters
If your printer supports it you may have letters like 'y' and 'p' set to print with descending tails.

Clear Edit
Get this stuff outa here! Start with a clean slate.

Up, Down, Left, and Right
Keeps them little bitties rolling. Shifty characters moved out of corners.

Reverse Echo
This is nifty but weird to explain.[3] Read the manual or play with the font called 'noecho'.

FONT DESIGN KIT
WITH COLOR CONTROL

FONTROL™

SOFTWARE PACKAGE

Table of Contents

CINIUM™
TECHNOLOGY, INC.
1150 Kane Concourse, Bay Harbor, FL 33154 (305) 861-8010/(800)235-2803

A wholly owned subsidiary of American Software Technology, Inc. NASDAQ Symbol: ASTK

A few words for the lawyers..

These are all trademarks of their respective companies:

Apple	**Atari**
C-64	**CBM**
CC-2064	**CIE**
Commodore Business Machines	**C. Itoh**
Epson	**FX-80**
Koala	**Koala Pad**
Microtek	**MSD**
Prowriter	**Toshiba**
Wico	

And a few words for the users..

Toss this book in a corner, read it only if you must. Chapter 1 is a tutorial covering most of the capabilities and features of Fontrol and Color. The rest of this manual is reference for programmers and similar crazies. *Of course, our hearts and souls are in this manual and if you don't memorize every precious word we'll probably commit Honorable Hara-Kiri.*

Chapter 1

Introduction

1.1 Overview

Fontrol is used with the Commodore C-64 computer to create new fonts (character sets). The fonts can be used on your display with other programs. The fonts created with Fontrol may also be loaded into printers connected to the C-64. This allows them to print special purpose fonts like the APL font or Commodore's graphic characters. Epson FX series Printers and the C.Itoh 8510[1] are supported by the font downloading programs. Several utilities have been included to allow the user to easily write programs which send fonts to other printers. The fonts are stored on disk as 'memory images' making access to them simple and painless. This means they may be easily utilized by other programs to create special effects.

Fontrol's editor is visually oriented, fast and very easy to use. The editor will, optionally, make good use of a wide variety of plug in controllers including joysticks, touch pads, track balls and paddles. When using a controller with the editor the keyboard is almost unnecessary, since almost all functions use the 'point and press' command style which has become so popular with the advent of mice, track balls, touch pads and touch screens.

---

1. The C-Itoh printers require the optional 2K buffer installed.

Overview

To use the Fontrol package you must have a CBM C-64 computer and a compatible disk drive. Disk drives which may be used include the 1540,1541,2040 and 4040, all from CBM. Other manufacturers drives will also work fine if they are compatible with 1540/4040 format disks. If you want to print your fonts on paper you will also need a printer supporting 'user definable fonts' like those mentioned above.

The program Color has been included in Fontrol. This utility program allows you to adjust the display colors of the C-64 to your own preferences. Once this has been done, Color will thwart any attempts by other programs to change the screen or character colors, essentially locking in the colors you've chosen. Color works as a background process and is compatible with all BASIC programs and most machine language programs.

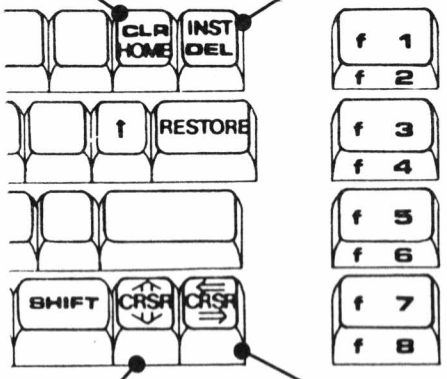
The font files are 4160 bytes long. In each file a total of 512 characters are stored. Only 256 may be displayed at one time. This corresponds to Commodore's convention of an uppercase/graphics and alternate upper/lowercase font. There are no restrictions, however on using each set any way you please. In fact if you need the extra characters, the second 128 characters of each set, normally printed in reverse field, may instead be special characters printed in normal video. The extra 64 bytes in the file which some of you may be wondering about are descender data. this allows us to send true (unlike the CBM screen font) descending characters to those printers which support them. Proportional spacing in printers is also supported with the width calculations being completely automatic.

1.2 Keyboard

If you have a controller of some sort: Plug it in and skip this page. Most controllers push the cursor around in a sensible way, You won't need the keyboard much. The cursor keys all behave normally, moving the cursor where you would expect them to. The shifted motion functions (up & left) have been duplicated on the CLR/HOME and INST/DEL keys for those who would rather stretch than shift.

CLR/HOME
Does cursor up.
Duplicates the
<SHIFT><CRSR>

INST/DEL
Does cursor left.
Duplicates the
<SHIFT><CRSR>



Cursor up/down
Moves the cursor down if unshifted, up if shifted. use to move between functions on the edit screen, change colors in Color.

Cursor left/right
Moves the cursor to the right, or to the left with the shift key.

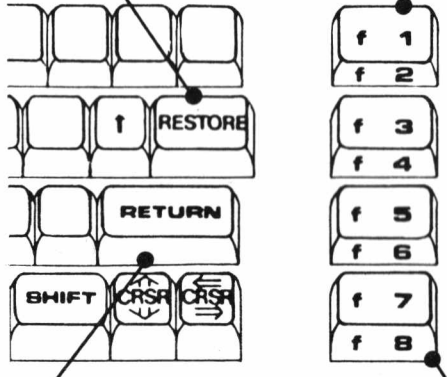
Keyboard

'DO' keys

Most functions of the editor work by putting a pointer in a window and telling the program to 'DO' what the window says in it. The button on your controller gives a DO command, so do these keys.

RESTORE
Gets you out of edit and on to the print menu. Same as 'quit', more for the lazy.

F1
(Function key 1) does 'DROP' which causes the character in the edit window to be placed under the cursor in the font being edited. More on this under 'Drop'.

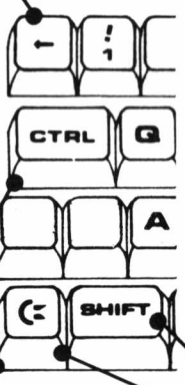


RETURN
Tells the editor 'do'. Whatever function the cursor is sitting over gets performed.

F7
(Function key 7) Just another 'DO' key. Easy to hit with the heel of your right hand.

These keys are not processed by the Fontrol editor, but are handled by Color or the C-64 Kernal.

Back Arrow
Turns off Color, redirects the irq vector back to the normal keyboard handler. You must have Color's menu bar on the screen for this key to work.



CTRL/CBM
Pulls up the color menu. These keys are patched through the keyboard handler in the kernal. They'll work even when Fontrol isn't running. More on this under 'Color'.

SHIFT/CBM
Changes between the uppercase/graphics font and the uppercase/lowercase font. This is the same key sequence that Commodore uses to choose between fonts.

The rest of the keys work only when the cursor is in a field which requires character stuff like filenames or drive numbers. A message will appear on the screen whenever the other keys can be used.

1.3 Controllers

Fontrol will use a wide variety of control devices. All controllers should be plugged into control port 2. We prefer touch pads, control is more intuitive and faster. Just try whatever control device you have, it'll probably work fine with Fontrol.

If your control thing has two buttons you'll probably be able to use one of them for DO, the other for DROP, try it.

The C-64 control ports are on the right side of the C-64. There are 2 control ports on your computer, port 2 is the 9 pin socket closest to the power switch and a little forward of it. If you'd like to know more about controller interfacing see the appendix on controller input.

Fontrol automatically senses changes in controller styles on the fly, so you may change controllers while the program is running.

1.4 Loading

Plug any control stuff you might care to use with the editor into port 2. These controllers are all optional, the editor will work fine without them. (But it'll be more fun with them.)

Turn on your C-64, disk drive and printer.

[Optional] Redirect I/O through custom drivers. If you have an IEEE-488, serial or parallel interface on your C-64 you should load and run any software they might require now. See the appendix on Bus Interfacing for more on this.

Put the Fontrol disk in your drive. and type:

LOAD "*",8 <return>[2]

or type:

LOAD "FONTROL",8 <return>

When the C-64 says 'READY', type:

RUN <return>

---

2. It's easier to remember the '*' but remember that this only works if you've just reset the drive. If you've already loaded a program from the drive before you put the Fontrol disk in, you'll get an error with the '*' because it tries to reload the last program it loaded.

Loading

This program loads several machine language programs including the Fontrol editor, some printer utilities, the screen utility 'Color' and CBM's wedge. Everything gets tucked into place by the loader which then passes control to the Fontrol editor. If all has gone well on the load you'll be looking at the edit screen. If not (unlikely, we hope) please turn to the appendix on loading problems.

There's a cursor blinking on the quit command now. This tells us that the editor will quit if we press the button on the controller or hit the return key on the keyboard.

If you're trying to send a font to a printer you're on the right track, just quit the editor now and you'll be in the printer menu. Go to the chapter on printer font downloading.

If this is your first session with Fontrol, you'll want to spend some time using the editor. Glance over the instructions on the keys and controllers Fontrol uses. Then go on to the Fontrol Editor Tutorial.

1.5 Editing

If you haven't loaded the Fontrol programs yet, please go to the previous section 'Loading'. Come back here when you've got the program loaded.

This section presents a quick tour through the editor facilities. If you have a specific question about a Fontrol function you may want to flip to the reference chapters.

Edit

I suspect that you've already been wandering around the screen doing ridiculous things to the poor @ character that was sitting in the window. If not, grab your joystick/ paddle/ touchpad/ trackball/ keyboard (hereafter referred to collectively as 'controller') and move the cursor up to the edit window. (The window in the middle of the screen) When you're in the window the cursor will change shape from a block to a cross.

Press the button (or <return> key) a few times... Move the cursor in the window, press the button some more.

Starting over & Getting fresh Characters

Well, if you've ruined the @ enough maybe we should try to get it back to where it started. Find the box labeled 'UNDO' and move your cursor into it.

UNDO
CLR EDIT
GET NEXT

Whack that button! Is it fixed? Undo copies the character sitting up in the

undone window into the character bit map. this means that you get to re-edit the character from where you started last time. When you start to edit a new character, the undone window gets it too. Undo also moves the cursor back into the edit window, ready for more editing. This is a trait of the three commands sitting in the group, UNDO, CLR EDIT, and GET NEXT. They all assume that you'll want to edit the character as soon as you've done these things, so they set you up for it.

CLR EDIT (Clear Edit Window) I'll bet you already know what this one will do. If there's doubt in your mind try it. Move the cursor over the 'CLR EDIT', (the words will blink) then press the button.

Disk Directories

Lets move the cursor down to the 'DIRECTORY' window. Is the Fontrol disk still in the drive? If not stuff it back in. Hit the button. You can start, stop and slow down the directory display with the button on the controller. Once you've got some of the Fontrol Directory displayed,look for some files in the form 'XXXXX FONT' these are all alternate fonts that we can load.

DIRECTORY LOAD SAVE
FILE IS:
LOAD ADDRESS $3000
TELL BUS:

Loading a Font File and the

Font File Names

The editor always needs a filename before it can fetch or save your fonts

from disk. Move to the empty space to the right of 'file is:'. The message 'keyboard is active, you may type now' will appear on your screen. Type a font name from the directory listing. 'computer font' would be a nice choice. Fontrol looks in this window for a font name to use when it loads and saves fonts.

Now that we have a filename set up, we're ready to load the font.

Move the cursor into the 'load' window and press the button. Since the load command is dangerous and can destroy work you've done but not saved, it will ask for confirmation. Press the 'y' key to start the load.

Choosing Set 1 or Set 2

Hold down the shift key and press the CBM logo key (it's on the lower left, with a stylized picture of Commodore's logo on it.) you've just changed between one font set and the other. Press it a couple more times (shift key too) and watch the 'set #' display on the top of the screen. This is the same sequence of keys that Commodore uses to select either set. Which set you choose to modify is completely up to you. We suggest that you use set 2 for fonts which include lowercase letters and set 1 for fonts which are uppercase and special characters only. This is to maintain some small degree of compatibility between your special fonts and the standard Commodore fonts.

Working on Chr$( )

At the top center of the display you'll see "working on chr$( )". This number does not refer to the ascii number of the character but, rather Commodore's

'screen poke value' of the character being edited. When building fonts for the screen we'll want to put the new characters in the same positions as Commodore uses. This means that 'A' will be chr$(1), 'Z' chr$(26) and so on. The printer portions of the program perform automatic translation of the font from the poke values to ascii so you'll probably want to keep fonts destined for the printer in the same order as screen fonts. More on this in the appendix 'Ascii vs. Petscii'

Fetching Characters to the Edit Window

If you look on the right side of the screen you'll see a block of characters. This is the 'character select window'. To select a character to edit: Move the cursor into this window. Place the cursor upon the character you wish to edit. Press the 'do' button (or <return> key). Try it now. The cursor and the character you picked are moved into the edit window. Notice that the 'undo' window has the character you've just selected in it. Once a new character is selected for editing, the previously edited character cannot be undone.

Drop

Move back into the Character Select Window. Good, now press the F1 key (or right button on most two button controllers). You've just dropped your first character. Why not move around and drop a few more? Drop will also let you carry characters between set 1 and 2. Especially handy if, like me, you tend to start an edit session in the wrong set.

Trash this font. or

Get this junk off my screen!

Sooner or later you're going to find yourself trying to edit a file which has become so butchered that the edit screen looks like an acid flash. To get a hint of just what this might look like, move the space character, chr$(032), into the edit window. Put a couple rows of lines in it, just doodle it up real bad. Look at that mess you've made, wait till Mother sees!

You can make a far worse mess by trying to load a program file as a font, that's when this command comes in real handy.

Move the cursor up to the extreme top left corner of the screen, over by the 'undone' window. There's no indication that we've put a command window there, it was gonna be a secret. You need to peg the cursor in the top left though. Press the 'do' button. The message 'Move CBM Font down from ROM' will be flashing on the screen. This command, like 'load' requires that you confirm your intention to destroy the font in memory. Press 'y' to tell the editor you want the CBM font back.

There are going to be times when you've fouled up a font so bad that you can't even find the cursor; this is why the 'Trash this font' sits up in a corner, you can get there even if you can't see what you're doing. Of course you may not be able to read the 'get ROM font' message either, but you'll probably know it's flashing.

The Shifty Fellows:

UP
DOWN
LEFT
RIGHT

Have you got a character in the edit window? If not, go pick one up. Anything will do as long as the edit window isn't empty.

Get the cursor into the 'up' window and hold down the button. Watch the image of the character you're working on in the character select window. Notice that even as you're editing the characters, the C-64 will use the new version on the display. Go ahead and try Down, Left and Right. These commands allow you to pretty up your fonts by adjusting a character's position after you're happy with it's appearance.

Get Next I found that many users of Fontrol would start editing at their favorite letters, do a few, then go to 'A' and step through the font alphabetically. We decided to let the machine remember the alphabetical order for us.

1.6 File Control

The windows at the bottom of the screen look to the C-64's peripherals. We'll move fonts in and out the editor by setting up these windows. If we want to talk to the disk drive, we can write in the 'tell bus' window. The commands we write here are sent down the 'disk error channel' (channel 15) when a 'DO' button is pressed in the window.

We're about to erase everything on the disk in the drive, pay attention!

Formatting ('NEW'ing) a Disk

Let's set up a new disk to store fonts on.

First get the fontrol disk out of the drive and put it away.

Find a new, blank disk and stuff it in the drive. Close the door.

Bring the cursor to the bottom line, next to the colon (:)

TELL BUS:_ *put the cursor here*

The screen says 'keyboard active...'

Type: **n0:my fonts,mf** <return>

<return> is the 'DO'button, the command is sent to the disk. The controller buttons send the commands as well. Fontrol will request confirmation of this command. Type 'y' to assure the system that you want this disk formatted.

The disk drive busy light should come on. Commodore drives take a couple of minutes to format their disks. When the disk is done we'll be able to store fonts and programs on this disk.

Saving a Font

Well, if you've been following along with us you have a freshly formatted blank disk in your drive. If not push one in or use a disk of yours that has at least 17 blocks free.

Check your filename in the 'FILE IS:' window;

FILE IS:_ *type a filename here*

Glance at the 'D' (drive #) and 'U' (unit or device #) windows, for most users they should say '0' and '8'. If not, please make sure they reflect stuff you've really got on your bus.

Move into the 'SAVE' window and whack the button. 1540/1541 users be patient, it'll finish soon enough.

When Fontrol is finished saving the file once, we'll save it again, same name, same disk. Do it now, press the button.

Fontrol will always check with you before overwriting a disk file. You can relax and not worry about blowing away your good fonts by mistake.

Talking and Listening to the Bus

Let's do something silly here. Open the disk drive door (leave it open). Now move down to the 'TELL BUS' window:

TELL BUS:_ *cursor here*

The screen says 'keyboard active...'

Type: **GO FASTER LITTLE 1541!**
(No, it doesn't work.)

Press <return> or a DO button.

You've confused the disk drive now, it's little red light is blinking. Move up to 'GET ERROR" and push a 'DO' button.

TELL BUS and

GET ERROR Let you talk and listen to any device hooked up to the bus. Remember that the 'U' (unit #) window determines which device you'll talk or listen to.

Don't forget to close the drive door. We opened it just in case the 1541 got confused by the command we sent.

Reverse Echo Please load up the font called 'reg/italic font'. This is one of the fonts on the Fontrol disk. This font is referred to on the cover as "NOECHO". Make sure you're in set #2. With reverse echo turned off you can build fonts like this. You might want to turn echo on (or leave it on) and edit a few characters while you're here. Made a nice mess of the font

yet? Notice in particular that the second space (chr$(160)) has been made into an underline character. If you don't do this, or leave this character reversed, you'll tend to lose the cursor on blank screens.

Quitting and

Getting ready to Print

There are two ways to exit the editor, the RESTORE key and the quit window. Both do exactly the same thing, moving you to a menu of Fontrol printer functions. if you quit the editor by accident, don't be concerned, your font won't be lost. Just hit the RETURN key, you'll return to the editor. The editor won't move the C-64 font down from ROM if it detects a good font in RAM.

Additional information on all the editing commands can be found in the reference sections of this manual. If you would like to go on to the print tutorial you'll need to quit the editor. The 'QUIT' window or the RESTORE key will both move you to the printer menu.

1.7 Printing

Before you start this section you should have completed these preliminary steps:

1. Connected a C-ITOH 8510 or Epson FX-80 to your C-64.

2. Tested the printer connection by printing something on the printer.

3. Loaded and run the Fontrol program, which left you in the editor when it finished.

4. Quit the editor (RESTORE key) leaving you looking at the menu "FONTROL PRINTER SETUP".

FONTROL PRINTER SETUP

USING SET [#] OF [font name]

PRINTER IS [printer name] AS DEVICE [#]

SET WILL [not] BE REORGANIZED

FUNCTION MENU

(P) CHANGE PRINTER
(D) CHANGE DEVICE CODE
(R) CHANGE reorganize
(F1) SEND FONT TO PRINTER
(F3) END PROGRAM

PRESS RETURN FOR EDITOR .

The top of this screen describes the current printer setup and details of the font in use. Each of these

things can be changed using the keys listed on the bottom of the screen.

(P): CHANGE PRINTER TYPE

Fontrol knows how to transmit it's fonts in formats that EPSON and PROWRITER printers can understand. Use the 'P' key to switch between the two types of printer.

(D): CHANGE DEVICE CODE

Changes the device number (unit number) of the printer. The default is device 4. Use whatever device number your printer interface is set to. Device codes 4 - 7 are allowed, Each time you press the 'D' key the number will change by one.

(R): TURN REORGANIZE ON/OFF

Determines whether the font will be rearranged into standard ASCII format (Reorg. on). Almost all users will want to leave Reorganize turned on. With reorganize off the font will be sent as PETSCII (half-ASCII).

(F1): SEND FONT TO PRINTER

Pressing F1 will send the font to the printer.

(F3): END PROGRAM

Terminates Fontrol, returns to BASIC. Leaves all the programs and font intact. You may restart by typing run.

RETURN key To return to the editor, just press the RETURN key.

SHIFT/CBM Selects set 1 or 2 to transmit to the printer, the set displayed will be sent.

Printer Differences

Fontrol's actions during font transmission are determined by what type of printer is selected.

Epson FX-80 If you have an Epson printer, the font will be sent to the printer, a character at a time. The command line at the bottom of the screen will reflect what character is being sent and what ASCII value it is being assigned to it.

C-ITOH 8510(Prowriter)

These printers require that font programming be the first commands to the printer after it is turned on. There doesn't seem to be any way to send a software reset to Prowriters. So . . .

Fontrol will ask you to turn the printer off, wait 5 sec. and turn it back on. Turning off a device that is connected to the computer's serial bus sometimes causes the system to 'hang'. Recovering from this can be tricky, try the RUN/STOP key or the RUN/STOP and RESTORE keys. If they work, they will leave you in basic. Type run to re-enter FONTROL.

The serial bus problem is a pain. There is no way around it. Fortunately, only once in a while will it happen. It's a good idea to always save your font before you try to send it to the Prowriter.

After the bus connection is established, Fontrol will send the font, and the command line will reflect what character is being sent, as described above.

NOTE for Prowriter owners... *You must have the optional 2K RAM buffer installed in your Prowriter. Otherwise there is no place to store the new font. This is a 2K by 8 bit RAM in a 24 pin package. Toshiba calls it a TMM2016, other manufacturers use similar names. In our Prowriter this RAM just popped into a socket at DIP position #16.*

Other Font-Programmable Printer

Well, I was gonna be clever here...

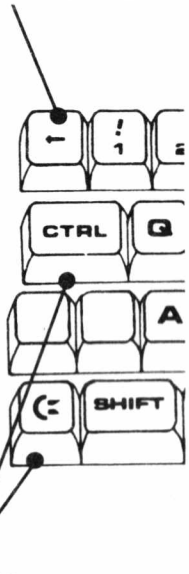
Several manufacturers are making printers labeled 'Epson Compatible', they may well be compatible in their font programming command syntax. Try to test them with Fontrol before you buy.

In order to send a font to another printer you must modify a copy of the Fontrol program. There is a remote (very remote) possibility that the Epson or Prowriter drivers will work on your printer. Try them for laughs. You'll probably make the printer crazy and will have to reset it before it prints again. There is an appendix on building downloaders in the back of this manual. This appendix tells how to add new printers to Fontrol. We suggest you aquaint yourself with this information before you buy a font programmable printer.

1.8 Color

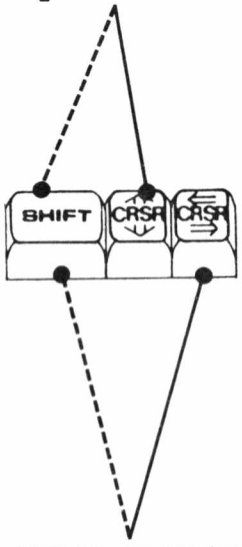
Have you loaded and run Fontrol? If so, Color is now part of the 60 hertz interrupt process in the C-64. This means that Color will continue to work in the background until the C-64's irq (interrupt) vectors are changed or you turn off color.

Back Arrow
Turns off Color permanently if pressed when Color's Menu is displayed.



CTRL/CBM
Press these together to display the Color Menu at the top of the screen.

CRSR/UP-CRSR/DOWN
Changes color of selected item: screen, border, etc. Or calls User's up or down CRSR functions, if they are installed.



CRSR/LEFT-CRSR/RIGHT
Moves cursor in Color's menu. Allows you to select item in menu.

Hold down the CTRL (control) key and press the CBM key. Color's menu bar has appeared at the top of the display:

Color's Menu
USE CRSR KEYS: SCREEN CHARS USER BORDER

Let's change the display colors now. Press the cursor down key a few times, notice that the color of the screen changes each time you do this. Use the cursor keys to select Colors for the Screen, Characters and Border. As you flip through the colors you'll see a great difference in contrast between the characters and the screen.

Just how any character/screen color combination looks on your Television is unpredictable. This is why we made Color, and specifically why Color is a background process. **Every time we set the colors the way we wanted them, some silly program changed them!** (Rude, Eh?) Invariably the colors the program chose looked awful on my monitors. Once you've got color running, those programs are not going to change a thing.[3]

When you are satisfied with the display colors just press any key that Color doesn't recognize. (The space bar is always easy to hit.) Color will restore the first line of the screen and hide itself again.

Color may be turned off, that is disconnected from the interrupts; Call Color (CTRL/CBM) then press the left arrow key. Color will remove itself from the interrupt chain. You may turn it back on again with SYS 49152.

A handy tip; Color can be used to start and stop program listings. just call Color (CTRL/CBM), when you want the scrolling to stop, press any key to restart the listing.

---

3. There are exceptions, See the appendix on color for fixes for programs that redirect irq vectors themselves.

Using Color Without Fontrol

There is no need to load all the Fontrol stuff if you just want Color in the C-64. To load Color alone:

1. **Load the program called 'COLOR $C000'.** It must be loaded in place (at the load address in the disk file) so the load command is:

   LOAD "COLOR $C000",8,1

   or if the wedge is in place:

   %COLOR $C000

2. **Use the SYS command to call Color** This is so Color may patch itself into the interrupt vectors:

   SYS 49152

That's all it takes. The file 'COLOR $C000' can be copied onto your favorite work disks if you like so that you won't have to shuffle so many disks around.

A user patch has been included in Color so that you may install machine language routines of your own. The appendix on Color includes a section on this, and on the C-64 interrupt routines.

Chapter 2

Reference

2.1 Editor

Alphabetical reference to all editor commands and functions.

Character Select

On the right side of the edit screen is the Character Select Window. This block displays a full Commodore font. Either set #1 or set #2 is displayed. When the cursor is being moved around inside the character select window, You may either drop or select characters.

The 'do' Key on your controller picks up the character under the cursor for editing. This button immediately puts the cursor in the center of the edit window with the character that has been selected.

The 'drop' key (F1 or right button) will put the current character (the character in the edit window) down at the cursor position.

CLR EDIT Clear the edit window and prepare to edit. The cursor is placed in the center of the edit window. Undo cancels this command, placing the

previous image of the letter back in the character select window.

D_ Represents the current disk drive. (0 or 1) DIRECTORY, LOAD and SAVE use this value in their commands. The editor will only accept '0' and '1' in the drive number. any other number will be rejected and when you move out of the field it will be reset to zero.

DESCENDING YES/NO

Just below the reverse echo window we'll find descender select. This Window does not appear on the illustrations. Sorry, printing deadlines hung us. When you press 'do' in the descender window, You're toggling the descender bit for the character in the edit window. descenders are only relevant to those people who are editing for printers. If you are building a character set that is going to be transmitted to a printer, and that printer knows how to handle descenders, then you can set the descender bit true for any character that you want to hang below the rest of the line. These typically would be the lower case characters g,j,p,q and y. If the descender bit is set for these characters they will be printed one dot row lower on the printer than the regular characters are. This gets the tails way down below the line and makes them more attractive.

DIRECTORY Fetches a disk directory, and displays it in a window on the screen. The drive number and unit number to the right control which device's directory will be displayed. These default to drive zero and unit 8, values appropriate for CBM disk drives.

DOWN Move the character being worked on in

the edit window down one pixel row at a time. Edge wrap is supported so that no character information will be lost if the character is shifted off the edge. Shifting a character around in it's display position will often place it better in relation to other characters on the display.

EDIT Characters are created and modified in the edit window. Each press of the button flips the bit currently underneath the cursor, so that pushing the button twice turns the pixel once off and then back on again or vice versa. Any character can be moved into the window using either GET NEXT or the Character Select Window.

FILE IS: The current file name. The text between the two quotes is the file name that Fontrol uses when loading and saving files. Leading and trailing blanks are stripped off the file name in the window, e.g. " pet font " is saved (and searched for) as "pet font"

GET ERROR Reads the error channel of the selected device (unit # sets the device) The error message is displayed in the status line.

GET NEXT Fetches the next character from the select window for editing. Increments the character string value pointer, 'WORKING ON CHR$(...)' seen at the top of the screen and places that character in the edit window. The cursor is placed in the middle of the edit window.

LEFT Move the character being worked on in the edit window left one pixel row at a time. Edge wrap is supported.

LOAD Loads a font file into the edit

buffer. The font files are always loaded into the editor at 3000hex.

LOAD ADDRESS This is the load address written to the font file when it is being saved. Only the first 2 digits of this field can be entered since the C-64 must keep fonts on a page boundary.[4] If you want access to both fonts (sets 1 & 2), you must specify a load address on a 2K boundary. More on this in the appendix on using fonts.

Remember that the editor completely ignores the load address and will always keep its font at 3000hex. This load address is strictly for the use of other programs that may want to keep their fonts elsewhere in memory.

OF SET ( ) The set number reflects the numbers that Commodore has assigned to the two different character sets. Set number one is upper case and graphics corresponding to the original pet character set. Set number two contains upper and lower case and a few graphic characters.

Since you are building your own characters you can make anything you like of these. It would probably be wise to echo Commodore's protocol in building your own character sets and put them in the set 1 position if they are primarily graphics; and the set two position if they include lower case

---

4. 6502 address space is divided into 'pages' of 256 bytes. In a 4 digit hex address the last 2 digits address the individual bytes in a page, the first 2 digits select 1 of 256 pages. 256 * 256 = 64K. Zero page is special and much used because it can be addressed with 1 byte.

letters. This of course is completely at the option of the user.

QUIT Leaves the editor and brings you to to the printer setup display.

REVERSE ECHO ON/OFF

Reverse echo dictates whether or not the editor will automatically copy changes in a character to the reverse field character that matches it and vice versa. If for example you are editing the reverse field A, and reverse echo is on then, the regular A will track the reverse field A and vice versa. Give this a try. It is easiest to just watch what happens when you switch reverse echo off and on. Turning Reverse Echo off gives you the option of having a total of 256 characters displayed as opposed to the normal 128 characters plus their reverse equivalents. The font 'REG/ITALIC FONT' which was to be called 'NOECHO' (see the package), demonstrates the use of this feature.

RIGHT Move the character being worked on in the edit window right one pixel row at a time. Edge wrap is supported. See also DOWN, LEFT, and UP.

SAVE Save a font file on the disk drive.

The first two bytes written to font files are their load address. This address is taken from the screen when the font is saved. Fontrol treats the font files like program files, loading and saving them as binary images. If your program can't use a font at 3000hex, a common problem because this is right in the middle of basic, the load address can be changed. When the editor saves the file it will be saved to reload at the address specified.

Fontrol, when loading a font, always loads at $3000hex, ignoring the load address (a documented feature [bug]). Therefore, be careful to always check the load address on the screen before saving a font with a special load address. If a font that was working fine in your program suddenly turns to trash, it's probably loading in the wrong place.

TELL BUS Used to transmit commands to any device on the bus. The unit number dictates which device gets the command. The normal DOS command syntax is followed. Your disk manual will give you more information about this. Remember, you may tell the bus anything through this command that can be said through the wedge or PRINT# commands.

U The UNIT #, or device address. Commodore started calling IEEE-488 devices 'units' with the advent of their DOS 3. I've followed that syntax here, hope It's not to weird. The largest allowable unit number is 31, the smallest 4.

UNDO Copies the image in the UNDONE window to the edit window. Sets the cursor in the edit window.

UNDONE This is a copy of the character that was most recently loaded into the edit window. The character you'll get in the edit window if you do 'UNDO'. It remains in the window until a new character is selected for editing.

UP Move the character being worked on in the edit window up one pixel row at a time. Edge wrap is supported so that no character information will be lost if the character is shifted off the

edge.

WORKING ON CHR$( )
Tells the user the screen poke value (in decimal) of the character that is currently in the edit window.

2.2 Building Downloaders

A guide to writing your own download module for printers not supported by the modules supplied.

By now you should be adept at printing normal text with your printer, and should have a working knowledge of BASIC. You will need both of these to get through this chapter.

Font bit Maps

All printers, terminals and computers that generate text in any way have one basic thing in common; they all have a stored image (bit map) of the characters they know how to print somewhere in their memory space. When we ask such a device to print or display a character, it uses this stored image to decide how. To install a new font we simply change the image. Changing the C-64's font memory is easy because Fontrol and the memory in question are in the same machine.

Printer Fonts

Printers are another story, their character memory is much less accessible to a program. Loading a font requires you have a printer whose operating system is capable of receiving and using a user-defined font. If this is not true for your printer, give up now.

To help you design and implement a printer module of your own, we have included listings of our modules and explanations of how they work. In addition, several machine language routines are attached to fontrol that perform some of the more complex tasks that would take basic all day.

Before we look at those modules, we need to become more familiar with how characters are defined and stored in the C-64,as well in the printer.

Characters created by the editor are stored as a series of 64 bits each. One bit for each block in the edit window. These 64 bits are arranged as 8 bytes. Each byte is eight bits wide, and represents one row of the character to which it belongs.

Descender Flags

In addition, associated with each character is a bit in another area of memory that determines if that character has a descender. Descenders are the little tails on p's and q's etc. that hang down one row lower than the bottom of the rest of the characters. The effect of this bit being set on the character as it appears on the 64's display is nil. The descenders on the 64's display don't really descend. Look at the upper/lower set carefully and see if you can figure out what we mean.

Some printers have an extra print pin on their print head that's not used in printing normal 8 row characters. Such printers also have a descender bit stored with each character who's value determines if the entire character is moved down one dot position so that the 'tail' really is lower than the bottom of a regular character.

Proportional Spacing

All of the characters created by the editor are 8 columns wide and 8 rows high. Some printers support variable character widths, so that thin characters such as I's and L's aren't printed with unnecessary space between them. This is often called proportional spacing, and makes printed reports look more professional. How Fontrol deals with this is explained in the discussion of each of the printer support modules.

Character Rotation

Most dot matrix printers store their font data in columns, this is because the print head, typically with 8 pins, sweeps across the page printing a column of dots at a time.

Since the characters are stored in the C-64 as a series of bytes, each of which describes a row of the character, and the printer wants a series of bytes each describing a column of of the character, we must construct an image of the character that is rotated 90 degrees.

Doing the rotation of the character in basic would be a mess and would take all day. To make it possible to send a font in a reasonable length of time, we have included several machine-language routines in the editor that are callable from basic.

The routines are ROTLSB and ROTMSB,and their function is to translate character data from horizontally stored bits to columnar bits. They are so named because both routines rotate (rot) the character image in opposite ways. Rotmsb rotates the character in the direction of it's most significant bit (msb), rotlsb rotates the character the other way, in the direction of it's least significant bit (lsb).

Try changing the direction of the rotation by changing the system call sometime and see what happens.

All the 'peeks' and 'pokes' and 'systems' in this part of the program are part of using the rotate routines. Briefly,here is what happens:

First the characters are poked into an 8 byte work area, WI, 'Work In'. Then the rotate routines are called, they do their thing and leave the resultant data in another work area, WO, (Work Out) ready to be transmitted.

Epson's Font Programming

Lets look first at the Epson Module since it is the easiest to understand, modify, and also it uses a standard that may be imitated (more or less) by other manufactures.

The first thing Fontrol does is open a channel to the printer using the BASIC 'OPEN' command. Next, the program enters a loop that is executed 256 times.

Once per character it sends this sequence:

esc & 0 S E A C0 C1 C2 C3 C4 C5 C6 C7 0 0 0 0

Where esc is the escape character, chr$(27)
& is an ampersand, "&"
S E is the character's Ascii value (sent twice)
A is the attribute byte. (more later)
C0 - C7 is the character data, a column at a time.

Attribute Bytes

Before we can transmit the character, one more thing must be figured out. Earlier we mentioned descenders and proportional spacing, and now we must construct a byte that tells the printer how wide the character is and if it has a descender. This is called 'the attribute byte'.

Epson printers interpret the attribute byte as follows:

---- -XXX Bits 0-2 are the binary value of the column the character starts in. Fontrol always sets this to column 1 or 001.

-XXX X--- Bits 3-6 specify what column the character ends in. Fontrol counts the number of non-zero columns in the character, adds 1 and sets the bits to this value.

X--- ---- Bit 7 is the descender bit. Epson's logic is inverted on this bit, so 0 in this bit position means descender, 1 is normal. Zero tells the printer to use the bottom print wire and to shift the character down a bit row. For most characters this bit of the attribute byte should be 1.

To summarize the Epson interface:

256 character images are sent
For each image:

The ascii value is sent
The attribute byte is assembled and sent
8 bytes of rotated 'image data' are sent
4 bytes of '0' are sent (fillers)

When all of the characters have been sent, Fontrol sends another escape sequence to select the font it has downloaded, closes the channel and returns to the program that called it.

Co-Authors note: I went through all this because the Epson format strikes me as a sane one. Chances are that if you have a printer that accepts font data at all, and it's not an Epson or Prowriter, this module will be closest to what you need and consequently easier to modify or otherwise use.

Prowriter Summary

The Prowriter printer does not allow character by character font programming. It does, however, let you modify its internal ram area to some extent. This is accomplished by sending the printer various incantations that set pointers and mess with ram allocation in strange and wonderful ways. The Prowriter requires that the bit images of the new font be sent as a single long block. So, for the prowriter;

Control sequence, (diddles buffer pointers)
2048 bytes of rotated 'image data' are sent
Control sequence (sets character pointers to buffer)

Since the protocol involved is so convoluted and is probably not used by any other printer, it seems pointless to describe in gory detail, exactly what happens when you run the module.

Briefly then; First Fontrol redefines the size of the input buffer so that the printer does not try to write over the font data we are going to send it. The font data is loaded directly into the printer's ram. Then Fontrol redefines the location of font data. The font ends up in the 2k of expansion ram,

leaving the buffer the same size as it would be in a non-expanded printer. This is why Fontrol requires the extra 2k be installed.

If you read a prowriter manual, you might not find any reference to this in it. My copy may be an early one, later revisions may or may not include information on downloading fonts. If you bug the distributor, they may send you one (this is how we got ours).

The Good News?

In any case, if you are a seasoned hacker, you should have no problem modifying either of these modules. If you just turned on your first computer, don't panic, just give up. Most probably fall between these two extremes, and if you are careful and understand what's going on before you chop up Fontrol, you'll be ok.

Important addresses

Things you must know to create a new downloader;

```
ROTMSB = $58BA = 22714,
      sys 22714 rotates WI to WO (towards
msb)
ROTLSB = $58D8 = 22744
      sys 22714 rotates WI to WO (towards
lsb)

WI = $5823 = 22563 (poke 8 rows before sys)
WO = $581B = 22555 (peek 8 columns after sys)
```

Memory below $2900 is tight, so when modifying Fontrol you may want to delete the downloader module(s) you're not using. There are a lot of REM statements ending in 98 and 99 which just make the code more readable, these can be deleted as well.

Appendix A

Ascii vs. Petscii

The American Standard Code of Information Interchange (ASCII for short) was adopted as a standard code for sending information between two devices. It is a seven bit code providing 127 unique combinations, or characters. Regular ASCII provides upper and lower case character sets, punctuation and control characters (intended for remote device control).

PETSCII is an eight bit code that provides all of the above, as well as some graphics and inverse characters. The two are similar, but not identical. The net effect of all this is that a character set that works fine on the C-64's screen will do strange things on your printer. In order to get around this mess, the download code in Fontrol re-arranges your font as it sends it to the printer in the following way:

PETSCII VALUE		PRINTER RECEIVES
0 - 31	.....	96 - 127
32 - 95	.....	32 - 95
96 - 128	.....	0 - 32
128 - 255	.....	128 - 255

It's not impossible that you may want to transmit a character set unaltered. To do this, turn the REORGANIZE option off. All this is designed to be transparent to you, the user. If you need to write your own download module, however, you should be aware of it.

Appendix B

Loading Problems

If the edit screen hasn't appeared or the cursor isn't blinking after you've tried to load Fontrol:

1. Check your disk drive to see if the error light is on and reseat the disk in the drive.[5]

2. Check the connections between the 1541 and the C-64, we've never had much luck with those little DIN plugs.

3. List the loader program ('FONTROL'). If it seems to have loaded ok try the 'RUN' command a couple more times.

4. You might try loading and listing the directory with:

   LOAD "$",8 <return>

   then

   LIST <return>

   If this doesn't get you a listing of our files we're probably in trouble, try turning everything off, **get the disk out first**, and starting over.

---

5. Peek inside the disk drive, Timmy's garter snake crawls in there for the warmth. Completely fouls things up around here.

If none of this makes it and you find that your system is working correctly with other disks, you may have a bad Fontrol disk. Try it in a friends computer or the store where you bought it. Did you make a backup? Ah well, none of us do when we want to try a new toy out...Get in touch with us, we'll replace your defective disk.

Appendix C

Controller Input

Fontrol will use a wide variety of control devices, all of which plug into control port 2.

Paddle controllers, Koala pads and **joysticks with potentiometers** in them (like the Apple joysticks): Any pair of variable resistance devices can be connected to the paddle inputs of port 2. The fontrol editor reads the resistance of the two paddles and converts them to an absolute X and Y coordinate on the screen. If you've got two buttons on the controller you've plugged in: The left button will operate as the 'do' (return key) control. The right button performs the 'drop' command (same as F1 key).[6]

Joysticks which use switches to indicate direction: The Atari, CBM and Wico joysticks are of this type. The button on these devices will generate the 'do' command. That is, they produce the same effects as the return key and F7 key. Use the F1 key for 'drop'.

Trackballs: The Wico trackball has been tested with

---

6. This is true for the stuff we've tested. It depends upon the buttons being wired to the Controller port like the buttons on a pair of Atari paddles.

Fontrol. Works fine, but my fingers get tired. Other trackballs will probably work provided they generate joystick motions. The button on the trackball generates the 'do' command. Use the F1 key for 'drop'.

All styles of controllers plug into **Control Port 2.** Fontrol automatically senses changes in controller styles on the fly, so you may change controllers while the program is running.

Appendix D

Bus Interfacing

IEEE-488, serial and parallel interfaces usually need a special initialization program to direct the Kernal I/O through them. Interfaces which plug into the cartridge port such as the MSD CIE will require a system call to their ROMs to hook themselves into the KERNAL. (CBM's name for it's operating system) For the MSD CIE the command is 'SYS 57278'.

Other interfaces, often hooked to the user port, such as Microtek's CC-2064 will require that you load and run their software before loading Fontrol.

If you have such a device and they stop working after Fontrol is loaded, check the appendix on memory utilization. Fontrol usually won't overwrite such things. Color, loaded at CC00(HEX), occasionally will overwrite the drivers for these interfaces. Often the drivers can be put elsewhere or Color can be turned off.

We've tested most of these to our satisfaction, but there's a joker in every deck.(often two)

Appendix E

Color Notes

E.1 Interrupts

The 6502 microprocessor in the C-64 is able to respond to signals called 'external interrupts'. These signals can have many sources but the one we are concerned with, the '60 hertz timer interrupt', occurs every 1/60 second. Each time the C-64 detects the timer interrupt the CPU stops execution of whatever program it was doing , saves enough information to return to the program, and services the interrupt.

How does the 6502 CPU find the interrupt servicing program? A pair of memory locations are used to store the interrupt program entry address. These 2 memory locations, called the 'irq vector'[7], are found in the C-64's system storage RAM at 0315 and 0316hex (same as peek 789,790). Think of this address as the first link in a chain of programs which execute each time an interrupt occurs.

When Color is started (system 49152) it saves the address it finds in the irq vector, then stores it's own entry address in the irq vector. This means that Color has added itself to the top of the 'interrupt chain'.

---

7. The word 'irq' means Interrupt ReQuest, and harks back to the naming of the input line to the CPU which is used to signal a request for interrupt servicing.

Each time an interrupt occurs (60 times/second) Color checks the keyboard for it's special key combination, the CTRL/CBM keys. If it sees these keys, Color displays it's menu bar and waits for the user to make a selection.

If there is no key pressed, or the key isn't one that Color recognizes, Color checks the display colors and corrects them if they've changed. Then control is passed (by Color) to whatever program was at the top of the interrupt chain when color was started. This program will usually be the C-64's interrupt handler which does a number of important housekeeping chores including keyboard scanning and clock updating.

If Color stops working when a program is run, the chain has been broken. No harm will be done, but color has to be turned back on with 'system 49152'. The chain will be rebuilt with a link added; Color will be monitoring the display colors and the keyboard again. The offending program will still have it's link in the chain.

The key point in this discussion is that Color keeps the chain intact. Not all programs that redirect the irq vectors will do this. If you want Color in control while these programs are running, you may add the call to Color (sys 49152) to these programs, probably right after the first system call in the program.

E.2 User calls

Color includes a facility for extending itself in it's menu bar, the USER window. If Color's cursor is in the USER window and a cursor up key is pressed, a subroutine call to USERUP (see the listing) occurs. The same occurs to USERDOWN with a cursor down key. This allows you to add your own functions to Color. By simply inserting a jump to your own programs in USERUP and USERDOWN the interrupt facility may be extended. Be sure to end your programs with an RTS command to keep the stack straight.

```
; FRAGMENT OF COLOR ASSEMBLY SHOWING USER ENTRY ADDRESS'S
; IN COLOR $C000
;                C2CEhex = 49870
;                C2D2hex = 49874
;
C2CE- EA        USERUP      NOP     ;USER MAY INSERT JUMP ADDR
C2CF- EA                    NOP     ;FOR 'UP' KEY
C2D0- EA                    NOP
C2D1- 60                    RTS     ;NOT USED IF JUMP ADDED
                ;
C2D2- EA        USERDOWN    NOP     ;ENTRY FOR 'DOWN' KEY
C2D3- EA                    NOP
C2D4- EA                    NOP
C2D5- 60                    RTS
```

Appendix F

Using Fonts

Four things need doing to make this work and keep the C-64 sane, You will need to include variations of these modules in your own programs if they are to find the new character sets. The tables in chapter 3 of the C-64 programming manual include further information on the process.

1. Tell the Video Interface Chip (Vic) which 16k bank to address. All video stuff, the screen, characters sets, sprite memory, etc. must be in the same 16K block for the Vic to find them. To do this we need to set the 2 low bits of location $DD00 (SCRHI) to the XOR'ed value of the high order address bits of the bank location. Since that made no sense, here's a table:

Set bits 0,1:	Addressing range:
11	$0000 - $3FFF
10	$4000 - $7FFF
01	$8000 - $BFFF
00	$C000 - $FFFF

2. Tell the Vic which of 16 1K blocks to start the screen in. The 4 high bits of SCRLOW are used by the Vic as bits 10 - 13 of the screen start address. (remember bits 14 and 15, the 2 highest in 16 bit addressing, came from SCRHI) So.. We need to take the high byte of the desired offset (from the start of the 16K bank) and multiply it by 4. The result of the multiplication becomes the high nybble of SCRLOW. An example:

We want the screen at $5800. In binary, this is '0101 1000 0000 0000' so our 2 hi bits are '01', Exclusive ORing (XORing) this gives us '10' which is what the low 2 bits of SCRHI need to be set to.

Subtracting $4000 from $5800 gives us $1800, our offset into the block. $18 (we only need hi byte) times 4 = $60, so $6 (0110) is the hi nybble of SCRLOW.

3. Tell the Vic where to find it's font. The 4 low bits of SCRLOW are used by the Vic as bits 10 - 13 of the font start address. (Bit 0 is ignored though, so fonts always start on a 2K boundary.) For the font address then, we'll divide the offset by 4, giving us the low nybble of SCRLOW:

   We'll put the font at $6000, SCRHI will be as in the example above. this leaves us with an offset of $2000, $20 divided by 4 = $8 (hex math). The low nybble of SCRLOW is $8 (1000 binary).

4. Tell the Kernal (Operating System) where the screen memory is. The Kernal looks to HIBASE ($0288) for the screen start address (high byte only) when it uses the screen.

   Our screen's at $5800 so we'll just store $58 in HIBASE.

Caution *There is a bug in the Kernal relating to HIBASE. RUN/STOP/RESTORE does not reset HIBASE to $0400, though it does reset the VIC to $0400. If this happens, nothing will appear on the screen when you type, but you may see characters 'under' the cursor. The C-64 is still ok though, and typing 'POKE 648,4' will make the screen reappear.*

Here's an assembler listing and a BASIC program side

by side. They show the steps required to set the C-64 to use a RAM font and redirect screen memory. You may notice that SCRLOW (or SLOW) gets peeked and poked more than seems neccesary. If you want to set both font and screen memory simultaneously, a single poke to SLOW will work fine. We illustrated the program like this only because it's the more general process.

Assembler:

```
SCRHI    .DE $DD00      ;6526 CIA
VIC      .DE $D000      ;VIDEO CONTROLLER
SCRLOW   .DE VIC+$18    ;BITS 4-7 VIDEO BASE
                        ;ADDRESS, BITS 1-3
                        ;CHARACTER BASE ADDRESS
; SET VIC RAM BANK (VIC CAN ONLY ADDRESS 16K)
         LDA SCRHI+2    ;PORT A DATA DIRECTION REG
         ORA #%00000011 ;SET 2 LOW BITS
         STA SCRHI+2    ;OF PORT A AS OUTPUTS
         LDA SCRHI      ;SETTING 1 OF 4 16K BANKS
         AND #%11111100 ;CLEAR LOW BITS
         ORA #%00000010 ;SELECT BANK 1 $4000-7FFF
         STA SCRHI
; SET SCREEN LOCATION WITHIN BANK
         LDA SCRLOW
         AND #%00001111 ;CLR SCREEN BITS
         ORA #%01100000 ;POINT SCREEN AT $5800
         STA SCRLOW
; POINT VIC AT NEW CHAR SET ($6000)
; BITS 1,2,3 OF SCRLOW ARE CHAR SET POINTERS
         LDA SCRLOW
         AND #%11110000 ;CLEARS BITS WE NEED
         ORA #%00001000 ;POINTING TO $6000
         STA SCRLOW     ;(VIC WILL ADD 2 HI BITS
                        ;FROM SCRHI TO GET
                        ;EFFECTIVE ADDRESS)
HIBASE   .DE $0288 ;OS LOOKS HERE FOR SCREEN ADDR
;TELL OS WHERE TO FIND SCREEN
         LDA #$58    ;SCREEN'S AT $5800
         STA HIBASE
```

BASIC:

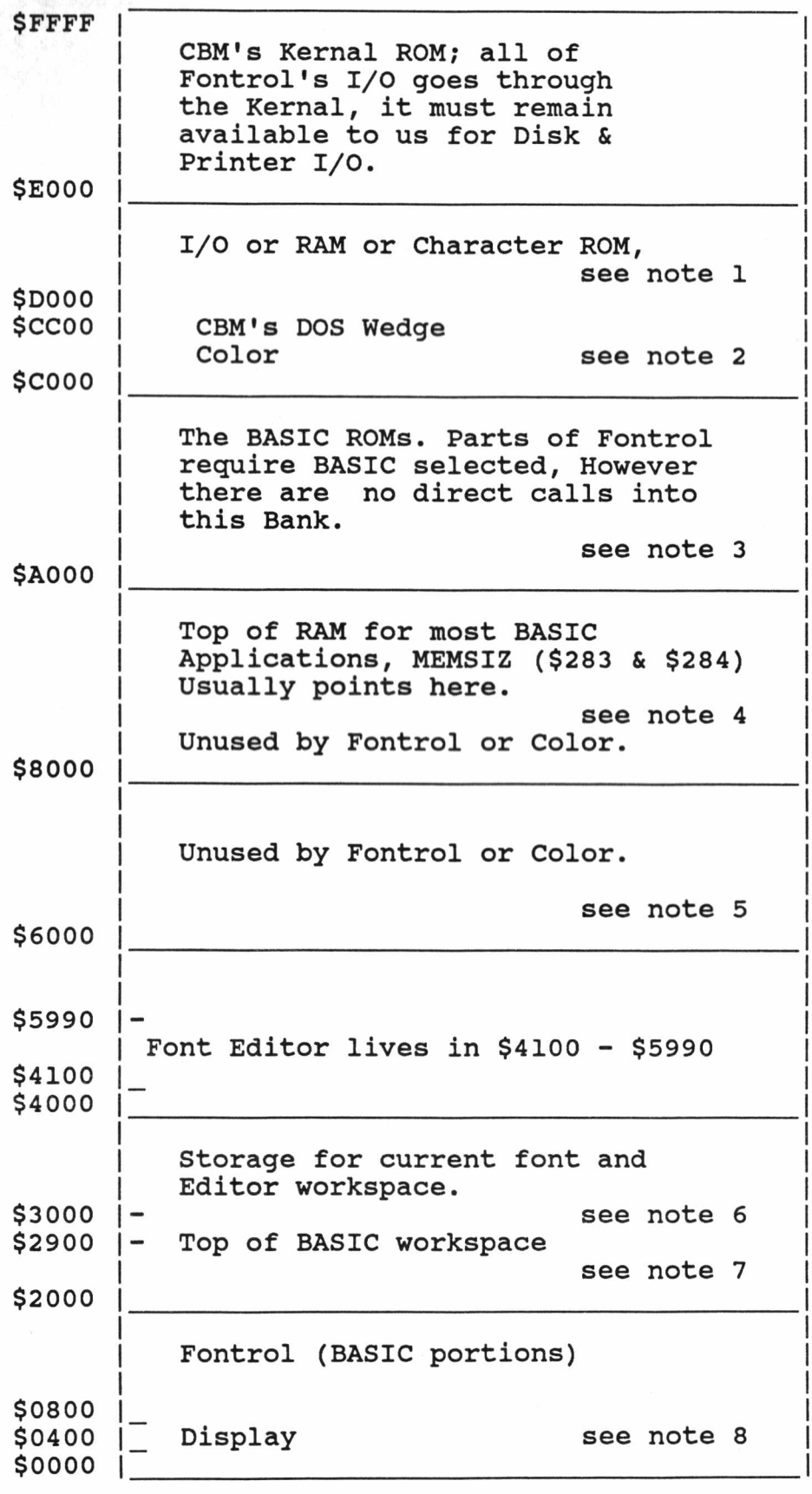
```
10 SCRHI = 56576
11 VIC = 53248
12 SLOW = 53272

13 A = PEEK(56578)
14 A = A OR 3
15 POKE 56578,A
18 A = PEEK(SCRHI)
20 A = A AND 252
21 A = A OR 2
22 POKE SCRHI,A

23 A = PEEK(SLOW)
24 A = A AND 15
25 A = A OR 96
26 POKE SLOW,A

30 A = PEEK(SLOW)
31 A = A AND 240
32 A = A OR 8
33 POKE SLOW,A

41 HIBASE = 648

42 POKE HIBASE,88
```

Appendix G

Fontrol's Memory Usage

This page intentionally wasted

Fontrol's Memory Usage



Note 1 This is the C-64's I/O space, Fontrol uses some of the VIC control registers as does Color. When the CBM ROM font is moved into Fontrol's storage area, the I/O is deselected and all interrupts are defeated. At all other times these addresses are connected to the I/O devices.

Note 2 CBM's Wedge lives here, the $C000 - $CFFF block is probably the busiest spot in all of CBM RAM, this 4K is used by a lot of other utility programs.

$C2E0-$CC00: This block is not used by Fontrol or Color, there's room here for the user callable functions of Color (2334 bytes between color and the wedge are unused)

$C000-$C2E0: Color lives here, the user jumps are at $C2CE and $C2D2. If you want to load something over color you must be sure to unlink color from the irq vector with the left arrow key. Otherwise the C-64 will probably crash when the irq entry code is overwritten. Color has been designed so that you may load Color over itself while it's running without crashing.

Note 3 The BASIC ROMs. The print functions and loader need BASIC to run, but we make no direct calls into this ROM. You may swap it out with supersets of BASIC in RAM and Fontrol will probably work fine.

Note 4 MEMSIZ, top of memory for the Operating System, $283 & $284, peek 643 & 644, is set to $A000.

This label (MEMSIZ) appears twice in CBM documents. The MEMSIZ at $37,38 (peek 55 & 56) is used by BASIC as top

of variables.

The other, at $283 & $284, is labeled "Pointer: Top of Memory for O.S." seems to be used by MEMTOP and the RS-232 handlers.

Note 5 The area between $5990 and $A000 is unused by Fontrol or Color and is available to other BASIC or machine language programs. (18032 bytes)

Note 6 $4000-$4040 is 64 bytes for storage of descender flags.

$3000-$4000 4k of storage for the character sets being edited. Characters are stored in the same format as in the C-64's ROM, with 8 consecutive bytes storing 1 character.

$2900$3000 Screen image storage, and miscellaneous Editor storage.

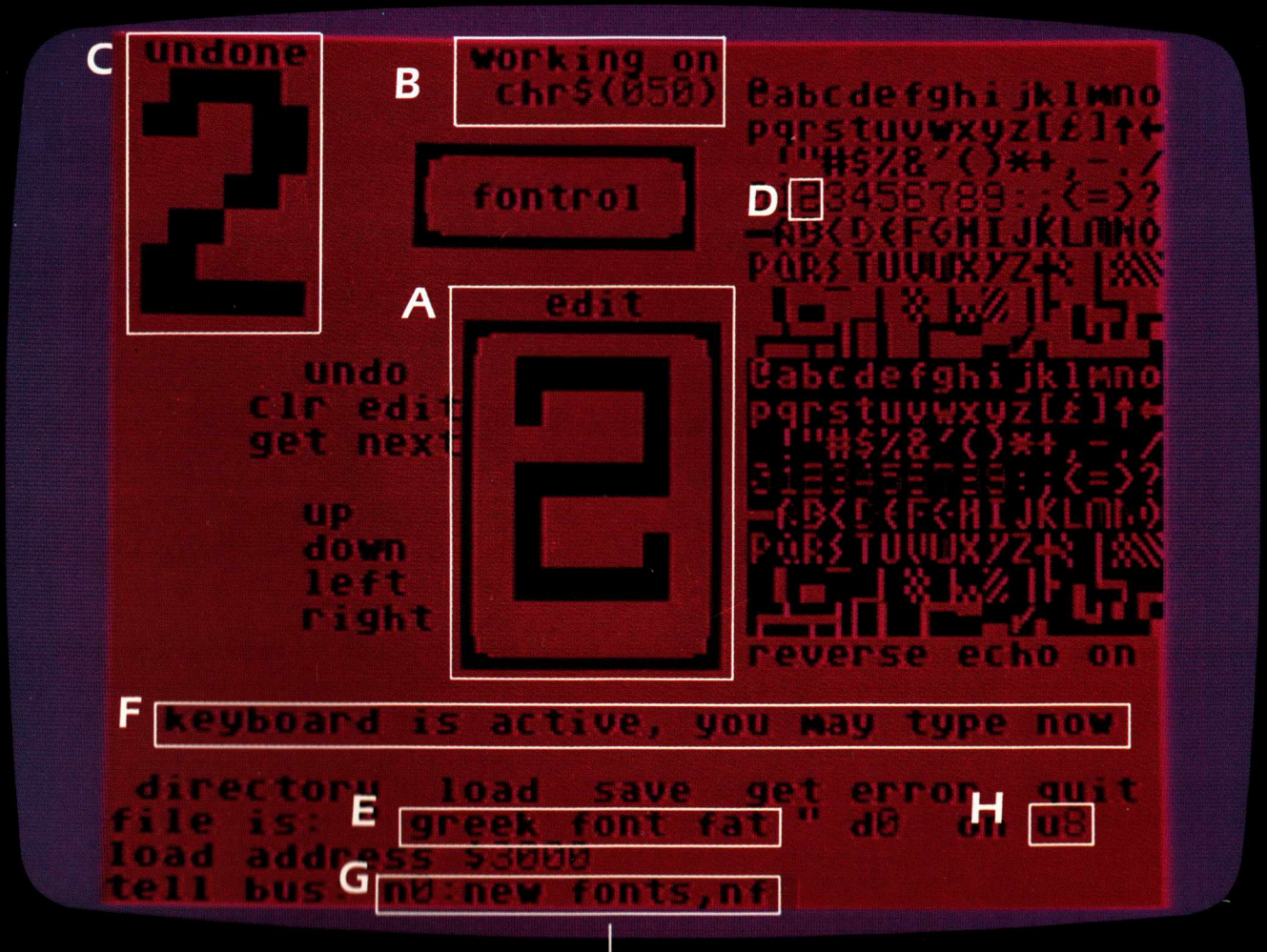
Note 7 The BASIC program Fontrol lives here. Top of BASIC memory is set at $28ff to safeguard the editor.

Note 8 0-$0800 C-64 system storage, screen display, stack, zpage. All zpage space used by the editor is restored on exit.

Index

An edit session underway...

Notice that we're working on the number '2'[A], the 50th[B] character in the set. The standard '2'[C] is in the undone window. And there is a rather rectangular '2'[A] being built in the edit window. If you look carefully you'll see that the new '2' is also visible in the character select block.[D] Looking in the device control window



we can see that the name 'greek font fat' is assigned to this font.[E] The message 'keyboard is active, you may type now' has appeared in the error/status line because the cursor is in the 'tell bus' window.[F] Whenever you may enter text at the keyboard, this message will appear. We're about to send a 'new' (format) command to the bus.[G] Since the unit number (on the right side) is set to 8, we know that this command will go out to device 8.[H] Notice that the command in the window has the same syntax as a DOS Wedge command.

Footnotes to Edit display

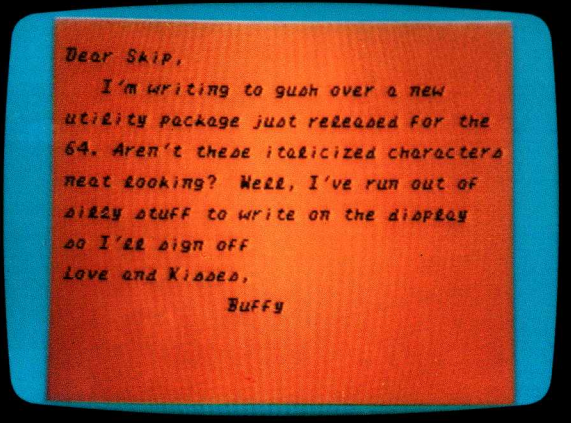
1. Fontrol is compatible with Joysticks (both Atari™ and analog types), paddles, Koala pads (our favorite pointer), and of course it works fine with the keyboard alone. (It's polite to point, but watch out for my eye, fella.)

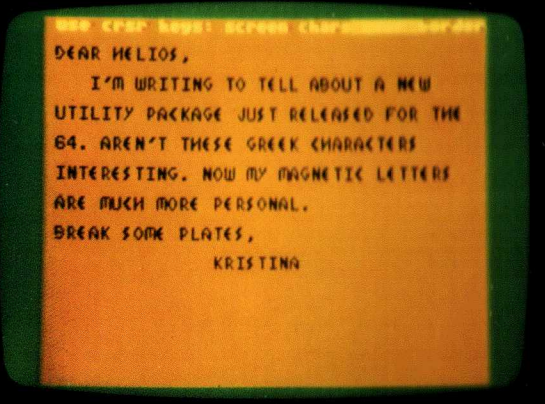
2. All 256 characters of one set (font) are displayed here, at any time a new character can be selected for editing by coming in here and picking up the character.

3. With reverse echo on, the characters exactly 128 positions apart will automatically echo each other, in reverse.

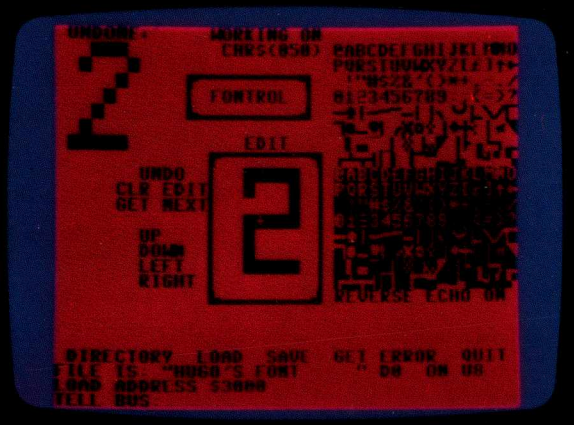
4. This window supports full communication with serial and IEEE-488 type devices, products such as the MSD CIE™ are fully compatible. Any command can be sent to any device. Destructive commands such as scratch and new (format) are verified before they are sent down the bus.

Fontrol™ Features, A few notes on why.

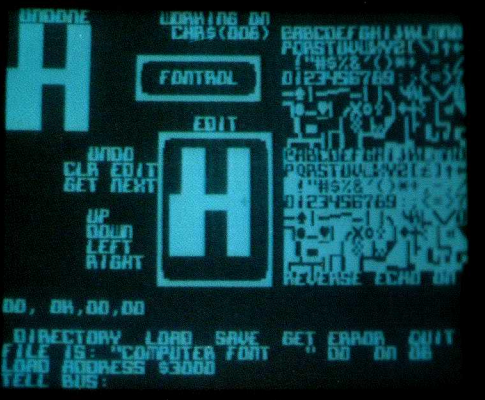




Buffy likes blues, but Kristina uses color to make a point. She's used the cursor keys to flip through her color choices.

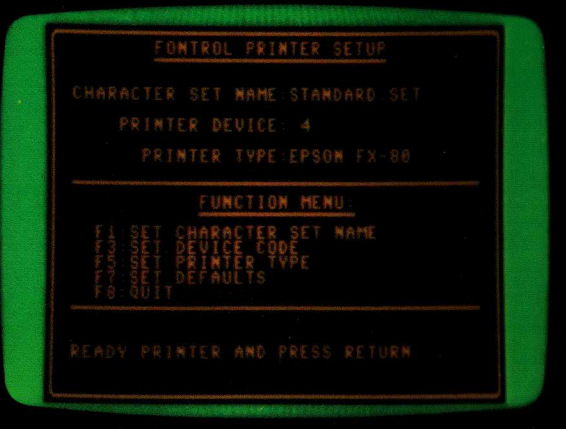


Even Buffy makes mistakes. The 'undo' command is a real lifesaver.

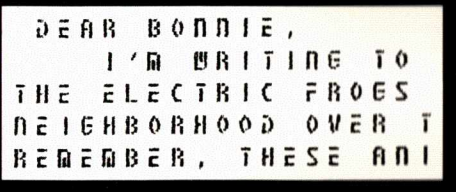


Make your computer look mean. (Scare your Bank.)

Commands like 'directory' and 'tell bus' make file handling a blast.



Printer control puts it on paper... Data to go.



1150 Kane Concourse, Bay Harbor Islands, FL 33154/(305) 868-4119